IN THE CLAIMS

1. (Original) A method for managing a code sequence, comprising:

processing a first set of sample values with coefficients from a first set of code sequence coefficients to determine first partial accumulation results during a first time step;

processing a second set of sample values with coefficients from a second set of code sequence coefficients to determine second partial accumulation results during a second time step;

processing the second set of sample values with coefficients from the first and second set of code sequence coefficients to determine third partial accumulation results during the second time step;

generating a lag result for a first sequence of sample values in response to the first and second partial accumulation results; and

generating a lag result for a second sequence of sample values in response to the first and third partial accumulation results.

2. (Previously Presented) The method of Claim 1, further comprising:

processing a third set of sample values with coefficients from a third set of code sequence coefficients to determine fourth partial accumulation results during a third time step; and

updating the lag result for the second sequence of sample values in response to the first, third, and fourth partial accumulation results.

3. (Currently Amended) The method of Claim 1, further comprising determining a synchronization point for the code sequence from the lag results for the first and second sequences of sample values.

4. (Previously Presented) The method of Claim 3, wherein determining a synchronization point comprises determining a lag result from the first and second sequences of sample values having the highest numerical value.

5. (Original) The method of Claim 1, wherein the first and second set of code sequence coefficients are contiguous coefficients from the code sequence.

6. (Original) The method of Claim 1, wherein the first and second set of sample values are contiguous sample values in a received sample.

7. (Currently Amended) The method of Claim 1, wherein to determine first partial accumulation results from the first set of sample values ~~and~~with coefficients from the first set of code sequence coefficients during the first time step comprises taking the products of the first set of sample values ~~and~~with the coefficients from the first set of code sequence coefficients.

8. (Original) The method of Claim 1, wherein generating the lag result for the first sequence of sample values in response to the first and second partial accumulation results comprises taking a sum of the first and second partial accumulation results.

9. (Previously Presented) A method for managing a code sequence, comprising:

accessing a first set of n coefficients in the code sequence and a first set of n sample values in a first sample sequence during a first time step;

processing the first set of n sample values with coefficients in the first set of n coefficients to determine first partial accumulation results;

accessing a second set of n coefficients in the code sequence and a second set of n sample values in the first sample sequence during a second time step;

processing the second set of n sample values with coefficients in the second set of n coefficients to determine second partial accumulation results; and

generating a lag result for the first sample sequence from the first and second partial accumulation results.

10. (Original) The method of Claim 9, further comprising:

processing the second set of n sample values with coefficients in the first and second set of n coefficients to determine third partial accumulation results; and

generating a lag result for a second sample sequence from the first and third partial accumulation results.

11. (Currently Amended) The method of Claim 10, further comprising:

accessing a third set of n sample values ~~in the sample~~ during a third time step;

processing the third set of n sample values with coefficients in the second set of n coefficients to determine fourth partial accumulation results; and

updating the lag result for the second sample sequence with the fourth partial accumulation results.

12. (Original) The method of Claim 9, wherein the first and second set of n coefficients are contiguous code sequence values in the code sequence.

13. (Currently Amended) The method of Claim 9, wherein the first and second set of n sample values are contiguous sample values~~ in the sample~~.

14. (Original) The method of Claim 9, wherein processing the first set of n sample values with coefficients in the first set of n coefficients to determine the first partial

accumulation results comprises taking the products of the first set of n sample values and the coefficients in the first set of n coefficients.

15. (Previously Presented) The method of Claim 9, wherein generating the lag result for the first sample sequence from the first and second partial accumulation results comprises taking a sum of the first and second partial accumulation results.

16. (Currently Amended) A method for managing a code sequence, comprising:

accessing sets of n contiguous sample values that include sample values in a plurality of sample sequences;

accessing sets of n contiguous corresponding coefficients; and

processing in parallel the sample values in each of ~~the~~a plurality of sets of sample values that are accessed with corresponding coefficients that are accessed, where each of the plurality of sets of sample values are processed during a different time step.

17. (Currently Amended) The method of Claim 16 further comprising generating lag results for each of the plurality of sample sequences.

18. (Currently Amended) The method of Claim 16, wherein each of the sets of n contiguous sample values is accessed at a ~~different~~unique time step.

19. (Currently Amended) The method of Claim 16, wherein each of the sets of n contiguous coefficients is accessed at a ~~different~~unique time step.

20. (Original) The method of Claim 16, wherein processing the sample values in each of the plurality of sets of sample values with corresponding coefficients comprises generating partial accumulation results.

21. (Previously Presented) A correlator unit, comprising:

a plurality of n sample sequence registers that store sample values from a plurality of sample sequences that are processed in parallel, the plurality of n sample sequence registers storing sample values from one set of sample values of a plurality of sets of sample values from the plurality of sample sequences at a time;

a plurality of 2n code sequence registers that store up to 2n coefficients from a code sequence; and

a processing unit that processes the sample values in each of the plurality of sets of sample values in the plurality of n sample sequence registers in parallel with corresponding coefficients in the plurality of 2n code sequence registers, where each of the plurality of sets of sample values is processed during a different time step.

22. (Currently Amended) The correlator unit of Claim 21, wherein the processing unit comprises an addition-multiplication tree having a plurality of specialized multiplexers that process contiguous sample values with their corresponding code sequence, and a plurality of adders, coupled to the specialized multiplexers, to accumulate results generated by the specialized multiplexers.

23. (Original) The correlator unit of Claim 22, wherein the addition-mulitplication tree comprises:

a plurality of specialized multiplexers; and

a plurality of adders.

24. (Original) The correlator unit of Claim 23, wherein each of the specialized multiplexers, comprises:

a multiplexer; and

a plurality of circuits that perform an XOR function.